Library information desktop application

Scenario B - CS106 Integrated Studio II

Nipun Goyal - 270249563

Contents

Introduction

We have been given a theoretical brief from a client who requires a desktop application to manage the in-house functions of a physical library, we've interpreted this as an application that allows the finding of books physically via the ISBN number and the loaning of books virtually from home, which is important in a post pandemic setting.

We aim to develop an application that can effectively manage all these functions including member management system, book transactions, cataloguing, and record-keeping and provide a comprehensive and user-friendly tool that can improve the efficiency of the library, both employees and its users.

In this report we'll outline our objectives, discuss our timeline roadmap, assign roles and tasks, define our process and toolset and complete the necessary documentation as part of the software development cycle.

This report is separated into three main phases:

1. Project Description and definition
2. Software Requirements analysis and specification (Planning)
3. UI Design can be done by using Figma or Adobe XD (UI/prototype Design)

Phase one - project definition

Goals and Objectives

Before undertaking this project, it is important to define clear goals and objectives. Goals refer to broad statements that describe what you want to achieve in the long-term whereas objectives are shorter term specific and measurable steps to meet your goals. These will be referred to throughout the software development cycle and tweaked and adjusted when necessary.

Goals

1. Improve the efficiency of library operations by streamlining and automating common tasks, such as cataloguing, member management, and book transactions.
2. Enhance the user experience for both library employees and members by providing a user-friendly interface that is engaging.
3. Increase the accuracy and reliability of library records by automating record-keeping and minimising manual errors.
4. Build a software application that is both dependable and functional, with the flexibility to expand and add new features as needed.

Objectives

1. Develop a user-friendly interface that allows library staff to easily manage and track member information, book transactions, and cataloguing.
2. Implement a database system that reduces errors and enables real-time updates to library records.
3. Build a search functionality that enables members to search and find books easily, based on various search criteria such as author, title, and genre.
4. Create a reporting feature that generates real-time analytics and reports based on data collected by the system.
5. Ensure the system is reliable and secure to prevent data loss or unauthorised access.

Project user requirements

1. Administrators and members should be able to log into the system.
2. The Administrator should be able to view books and members' information.
3. The Administrator should be able to add new books and modify the catalogue.
4. The Administrator should be able to add new members and modify their information in the system.
5. The system should log messages in a "due-date" file whenever the due date of their loaned books is nearby.
6. The system should log messages in an "overdue" file whenever a book is overdue.
7. Members should be able to view the catalogue along with availability.
8. Members should be able to pre-book the books.
9. The system should log messages in the "return" file whenever a book is returned.

Potential additional features

1. Automated email loan reminders to library members
2. Reserve books and have automated emails when back in-stock
3. Include book imagery and copy of blurb potentially using the Openbook Covers API
4. A mechanism for adding reviews and a 5-star rating system
5. Produce reports with daily, monthly analytics etc.

6. With the user survey we can ask him as to how long did it take him to go through just for a reference for the new readers.

7. Filter results based on criteria like language, author, year, adult/child, etc. (This should be easy to do with LINQ and a well-designed database.)

Timeline and Project Constraints

Within our timeline of eight weeks, we'll move through the documentation and development in tandem. An earlier half-way document submission of our current progress will be submitted in week four.

Our documentation will be an evolving (and ongoing) process. The WPF application design and development shall be built over six/seven weeks, to be handed in to the "client" on the 2nd July 23.

We've identified some potential constraints or roadblocks preventing us from meeting our goals:

- Encountering the unexpected while developing, such as technology "can do" issues (can it be done) or whether an idea - even technical - can be developed in the anticipated time frame involved.
- The shift to C# and using the WPF framework will present new challenges, we may need to change the project scope accordingly to meet deadlines.

Selected software development methodology

Agile is most appropriate for the project and it's a recommended method in light of industry standards. It's anticipated that there will be daily meetings, of approx. fifteen minutes. These meetings are to serve several purposes, including to "check in" with each other. More than one meeting may take place.

Development is expected to be a mix of planned and unplanned work, testing and review of work done to date. Daily meetings should assist with keeping focus and ensuring that client deliverables are being met or, if not, to discuss ways to inform the client of issues faced. Effort will be made to document major issues as they arise.

In comparison the Waterfall method is a linear, sequential based approach to the software development process, lacking in the ability to accommodate unexpected change or change of mind on the client's part. Each phase is designed to be forward moving, like with the action of walking down stairs - it's a one way action. For our application this methodology won't be flexible enough as we are expecting to make changes throughout the project as we encounter technical limitations.

Agile development cycle

The Agile framework involves planning a number of sprints that each meet a core objective such as “complete front-end user interface”, “complete backend database”, etc. Each sprint cycle incorporates the following steps that are repeated until the final product is deployed.

Reviewing sprints

At the end of each sprint the development team, scrum master and product owner will meet to review each sprint. A review agenda would include the following steps

Review Sprint Goals

Usually some time has passed since the sprint commenced so this step is to refresh everyone on what the starting objectives were.

- What was planned to achieve
- Roadmap elements that we wanted to target
- Milestones within the sprint

Status overview

- Finished and unfinished items
- Added or removed items
- Changed priorities
- How to address and continue with incomplete work.

Live demonstration

- Demonstrations of functionality, features and build progress
- Ask for feedback

Sprint statistics

- This would be applying statistical models to assess, measure and visualise progress.

Impediments

- Identify risks to project delivery and objectives

Propose solutions

Development tools

(collaborative tools such as version control systems, testing, prototyping), technologies, IDES, programming languages, etc., are used in the project.

Visual studio IDE

Visual Studio 2019 is the IDE (Integrated Development Environment) being used. The latest version, Visual Studio 2023, is not used due to upgrade issues (time taken and organisation policy).

LINQ - data structure querying tool

A data-store querying technology, LINQ, will be used. LINQ facilitates querying a database directly from C# as well as querying in memory data structures.

Insofar the database is concerned: In the event that LINQ is not sufficient for querying a database, custom stored procedures will be developed and a .NET library for accessing the database shall be used.

Behind the scenes LINQ creates SQL queries ‘on the fly’. If the query doesn’t perform well or does not return a technically accurate result, then a custom query will be authored (with a .NET library being used for access).

Database technology

The open source PostgreSQL database system will be used. For work requirements a cheap, cloud-based system is to be hired. Options are “ElephantSQL” or “Heliohost” or a Virtual Machine with PostgreSQL installed.

https://www.elephantsql.com/plans.html

https://heliohost.org/

This .NET connector will be used, mindful of its limitations: https://www.npgsql.org/. This is for C# database access.

Issues with a 3rd party app: It has been found from experimental work that the NpgSQL connector has some bugs and constraints. LINQ to Object practices may be used where needed to overcome limitations of the .NET connector (once data has been retrieved).

For example, a code sample for the NpgSQL connector was limited in its ability to be adapted (a bug was found after adaptation)). It was a code sample to get data from the database using C#. A workaround was found which ultimately would best be done using LINQ to Objects.

Figma

Figma is used to devise WPF screen mockups and to convey an idea of use flow.

GitHub

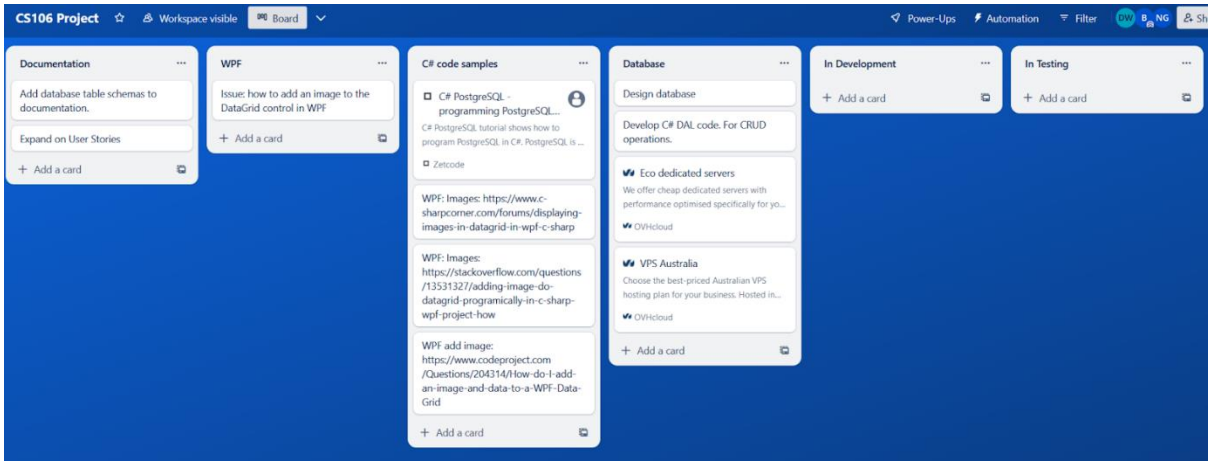
GitHub is used for source control management, C# and SQL code files, if the latter becomes necessary.

Google drive

A shared project folder will be used to host course and development materials.

Trello

For Project Management and to keep track of issues in general, the Trello online tool is used. It has been organised to reflect areas of work, such as Documentation, WPF, C# Code samples and Database. Other “cards” are In Development and In Testing.



Phase two - Software Requirements, research and analysis

Identification of functional and non-functional requirements.

The identification of functional and non-functional requirements is an essential step in software development that ensures that the software meets the needs of the end-users.

- Functional requirements define what the software should do or the functions it should perform to meet the users' needs.

- Non-functional requirements, on the other hand, describe how well the software should perform those functions, such as its speed, reliability, security, and usability.

Research - Eliciting requirements from clients

To identify the functional and non-functional requirements, we need to engage with our stakeholders, including the library staff and members, to determine what they need from the library information system. Once these requirements are identified, they can be organised, prioritised, and documented in a software requirement specification document. The SRS document serves as a blueprint for the software development team, ensuring that the software aligns with the stakeholders' needs and requirements.

The following methods have been used to collect this information to guide and define these requirements:

Interviews

We formulated a number of questions as a group then collectively took turns roleplaying potential responses from the librarian's perspective. We then selected what we thought would be the best responses.

Interview - Librarian at Auckland Central library

Interviewer: What are some of the main challenges you face in managing the library's collection and transactions?

Librarian:
One of our main challenges is keeping track of all the books that are borrowed and returned. We also have a lot of members who come in and out of the library, so managing their records and membership status can be time-consuming.

Interviewer: How do you currently keep track of the books and member records?

Librarian:
We use a fairly basic software system where we record all the book transactions and member details in a digital ledger. It's not very efficient, and we often have to spend a lot of time searching for specific records as there is no visual UI as it’s a text based command line. A barcode can be scanned on the book relating to the ISBN number that can then reference the book on our libraries database.

Interviewer:
What are some features you would like to see in a library information system that could help address these challenges?

Librarian:
It would be great if the system could automatically update member records and send out reminders for overdue books. We'd also like a search function that makes it easy

to find specific books and member details. And if the system could generate reports on library usage and book circulation, that would be very helpful for our planning and budgeting.

Interviewer:
How important is ease of use and accessibility for both library staff and members?

Librarian:
It's very important. We want to make sure that staff can easily navigate and use the system to carry out their tasks, and that members can access the catalogue and information they need quickly and intuitively. It's important that the system is user-friendly and accessible for people of all ages and backgrounds.

Interview - Member of Auckland central library

Interviewer:
What are some challenges you have faced when using library services in the past?

Library member:
As a library member, I have faced a few challenges in the past. Firstly, I have found it difficult to keep track of the books I have borrowed and when they are due back. Secondly, it can be frustrating when the library does not have the books I am looking for. Lastly, sometimes the library can be quite busy, which means I have to wait in line to borrow or return books.

Interviewer:
How do you typically go about finding books to borrow from the library?

Library member:
When looking for books to borrow from the library, I usually start by searching the library's online catalogue. I use the search function to find books on specific topics or by particular authors. If I can't find what I'm looking for, I will usually ask the librarian for help.

Interviewer:
What features would you like to see in a library information system that would make it easier for you to find and borrow books?

Library member:
I would like a library information system to have a more user-friendly search function that provides accurate and relevant results. It would also be helpful if the system could show me if a book is currently available or when it is due back. Lastly, it would be great if the system could recommend similar books based on my reading history.

Interviewer:

How do you keep track of the books you have borrowed and when they are due back?

Library member:
Currently, I keep track of the books I have borrowed by writing down the due dates in a notebook or setting reminders on my phone. However, it would be much more convenient if the library information system had a feature that could remind me when my books are due back or even automatically renew my loans if no one else has requested the books.

Interviewer:
How do you provide feedback or suggestions to the library about their services?

Library member:
I usually provide feedback or suggestions by filling out a feedback form at the library or by talking to the librarian directly. However, I would prefer to be able to give feedback and suggestions through the library information system itself, as this would make it much more convenient and accessible.

Other ethnographic research

Observing User Behaviour within a physical library

Observations can provide valuable insights into user behaviours, preferences, and pain points within the library setting, here are some observations I've seen when stopping by my local library.

- Users browse through bookshelves, flipping through pages, and examining books.
- Users seeking assistance from library staff, asking questions about where to find things.
- Interactions between users and library kiosks for borrowing books.
- Users searching for books using the same kiosks for searching the library catalogue.
- Users utilising study spaces and seating areas within the library.
- Users talking quietly and using mobile devices.

Observing User Behaviour on the Auckland library website

I interacted with the Auckland libraries website.

https://discover.aucklandlibraries.govt.nz/search

- User initiating a search, such as entering keywords, book titles, or author names.

- Use of search filters or advanced search options to narrow down the search results.
- Interaction with the search results page, including scanning through the list of books, examining book covers, and reading book descriptions or summaries.
- Clicking on a specific book to view more details, such as availability, location, or additional information.
- Engagement with features like "Related Books" or "You may also like" sections for exploring similar titles and topics

User stories and scenarios

User stories are useful because they provide a concise and user-centric description of a feature or requirement. They capture the "who," "what," and "why" of user needs, helping to align development efforts with customer value.

We have used the following format to create a number of unique stories for our varying stakeholders.

User story: As a [description of user], I want [functionality], so that [user benefit].

Acceptance criteria what needs to be fulfilled to meet the users needs and complete the story: Given [how things begin], when [action taken], then outcome of taking action].

User Story 1 - Member loaning a book
User story:
As a library user I want to take out (loan) a book with minimal effort so that I can borrow a book easily.
Acceptance Criteria:
1. The library user should be able to search for a desired book by title, author, or subject. 2. The system should display the availability of the book (whether it is currently on loan or available for borrowing). 3. The library user should be able to select the book they wish to borrow. 4. Upon verifying the user's credentials, the system should record the borrowing transaction and update the book's status as "on loan." 5. The user should receive a confirmation message indicating successful loan of the book, along with the due date for its return.

6. If the desired book is not available, the system should provide alternative suggestions or a waitlist option if applicable.

User Story 2 - Creating a library member account

User story:

As a library user I want to take out (loan) a book with minimal effort so that I can borrow a book easily.

Acceptance Criteria:

1. The registration form should have clear and intuitive fields for entering personal information, such as name, contact details, and library card ID.
2. The system should provide real-time validation and helpful error messages for any missing or incorrectly entered information.
3. The registration process should require minimal steps and not overwhelm the user with unnecessary or complex requirements.
4. After submitting the registration form, the system should promptly provide confirmation of successful registration

User Story 3 - Library member loan reminders

User story:

As a library user, I want to be notified by an email message when a book I want to take out is available or if a book I have is overdue so that I can remember to return books or loan a newly available book.

Acceptance Criteria:

1. The system should provide an option for the user to subscribe to email notifications for book availability and due date reminders.
2. When a book that the user wants to borrow becomes available, the system should automatically send an email notification to the user, providing details about the book and instructions for loaning it.
3. If a book borrowed by the user becomes overdue, the system should send an email reminder, notifying the user about the overdue book and providing information on returning it.

4. The email notifications should include clear and concise information

User Story 4 - Admin editing active members
User story:
As an admin, I want to view a list of active library members and add, remove, or disable accounts with immediate effect so that users can be managed or loaning privileges disabled.
Acceptance Criteria:
1. The system should provide an admin dashboard or interface that displays a list of active library members, including their names, contact information, and current membership status. 2. The admin should have the ability to search, filter, and sort the member list based on various criteria such as name, membership status, or registration date. 3. The admin should be able to remove or disable member accounts with immediate effect, preventing them from borrowing books or accessing certain library services.

Requirements classification and ranking

These will be detailed within our SRS documentation. This involves sorting our gathered information to extract and organise our identified requirements to better understand the overall structure of our library software system. We'll sort these into categories such as data requirements, user-interface requirements etc and assign priorities based on their importance and feasibility to the completion and meet the requirements of our project.

SRS Document

The SRS document typically includes the following information:

Introduction

- **Introduction: A brief description of the software, its purpose, and its intended audience.**

Application purpose

The application allows the public to browse books for borrowing out. It also allows librarians to manage what books can be browsed by the public as well as manage the public's access to the application.

Intended audience

The audience that will use the application is a mixture of the general public and librarians. All ages have to be considered, from teenagers to the elderly. The elderly may be technology challenged and so the user interface of the application might have to be built with this in mind.

Intended Use

The client is a library and the service is loaning books to people. To manage books that will be borrowed by the general public. To manage those that will use the system. In both cases to be as easy as possible to operate.

User Needs

For the end-user: An informative yet easy to use system for finding and then borrowing books. For the library worker: An easy-to-use system for managing and creating library users. And for adding and managing books that will be borrowed.

Assumptions and Dependencies

There are no known dependencies. Assumptions: The library user is computer literate and knows how to operate a computer. The need to educate the library user will be a responsibility borne by the library.

Requirements from clients

- Windows operating system.
- Minimal number of open windows.
- One interface for both users.
- Easy to use as best possible for librarians and the general public.

Scope

- **Scope: A definition of the scope of the software, including the features and functions it will provide, as well as any constraints or limitations.**

The scope of the application is very limited. A list of features can be found below. But to provide a brief summary: the app is for facilitating the issuance of books for loan to library members and for the management of its users and the books.

The UI technology is provided by a technology called WPF (Windows Presentation Foundation). As this is new to the development team, a moderate learning curve is involved. The development team will endeavour to lessen the impact of this on the delivery timetable by spending evenings and weekends experimenting with WPF technology.

Some visual features may not be implemented owing to the aforementioned understanding.

A cloud-based database system, Postgre, will provide the back-end functionality. Ideally a free hosting account will be found otherwise a super cheap one with monthly based payment will be used.

Functional requirements

- **Functional Requirements: A list of the software's functions, along with descriptions of how they will work and how they will interact with other parts of the system.**

Databases

Some of the functionality below requires a database system. A RDMS system called “Postgre” will be used and a design (as per earlier in this document) will be implemented to aid with the realisation of the below. Some time will need to be allocated for experimentation purposes.

Functions:

1. User login
2. User log out
3. Create a book entry and manage this (book available, book not available)
4. Create a user and manage this (manage login ability)
5. Search books based on a criteria (author, genre, etc).
6. View book detail
7. For librarian use: be notified when a book’s due date is close (say 3 days).
8. For the public: ability to pre-book a book (for when it’s available).
9. For the librarian: record when a book is returned.
10. For the librarian: record when a book is overdue.
11. 5 star rating for the book

Nice-to-have functions:

1. For the public: automated email loan reminders
2. Book reviews
3. For the public: SMS notification when a book has been returned (i.e.: is available).
4. Daily/monthly reports on various book metrics (e.g.: book popularity, area popularity).

Non-functional requirements

- **Non-Functional Requirements: A list of the software's non-functional requirements, such as performance, scalability, usability, and security.**

As this is a student project with non mission critical requirements,there are no serious performance, scalability or security issues to address. However, one area to address is usability.

User interface issue

Public service treatment matters for a library and usability issues that go with this environment count. For example, a number of library users are retired or elderly people. Eyesight considerations may apply. In practical terms a user interface would need to have big size text for button text and text statements.

Database design

We are creating a relational database whose design will be kept as simple as possible. Primary and Foreign Key relationships will be set up if permitted by the database system. This will ensure database relational integrity: data won't be deleted if it is protected, etc. All tables will have a Primary Key column called "ID". In some tables there will be a Foreign Key reference.

Users table:

ID (bigint)	Username (100)	Password (100)	First Name (200)	Last Name (200)	IsAdmin (bool)	LastLoginDate (date)	IsEnabled (bool)

Wishlist table:

ID (bigint)	UserID (bigint)	BookID (bigint)	Added Date (date)

Books table:

ID (bigint)	Title (200)	Author (200)	Summary (1000)	TimeToRead (bigint)	Rating (bigint)	New Release (bool)	GenreTags (100)	CoverImageURL (1000)	AvailableToLoan (bool)	DueDate (string)

Bookreview table:

ID (bigint)	BookID (bigint)	Rating (bigint)

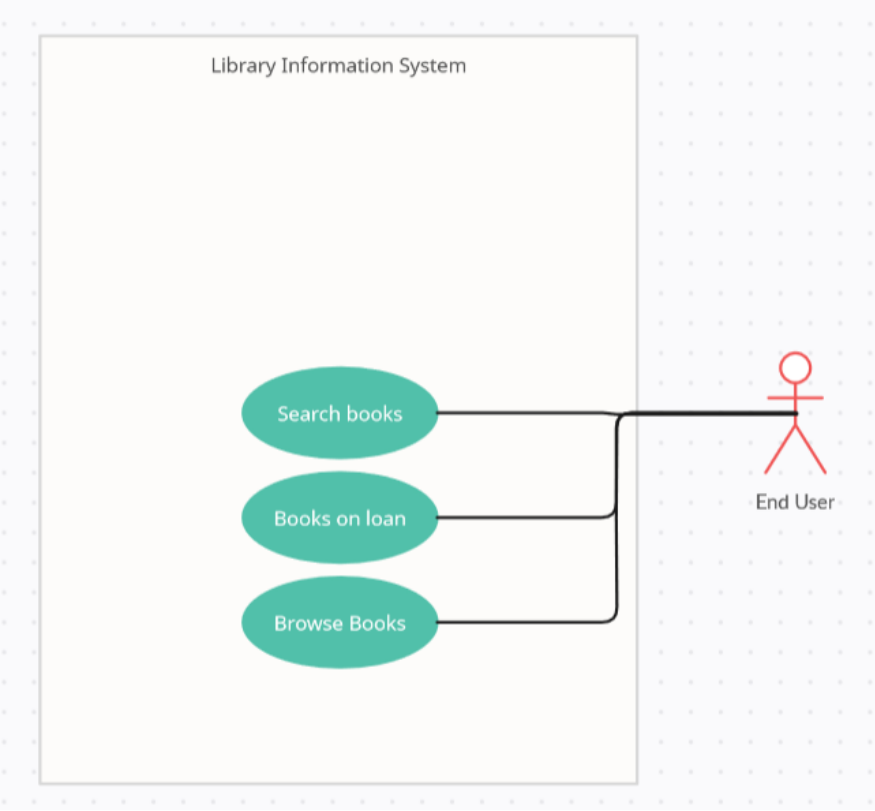
Loan table

ID (bigint)	BookID (bigint)	UserID (bigint)	IssueDate (date)	DueDate (date)	FineDue (bool)	FineAmount (decimal)

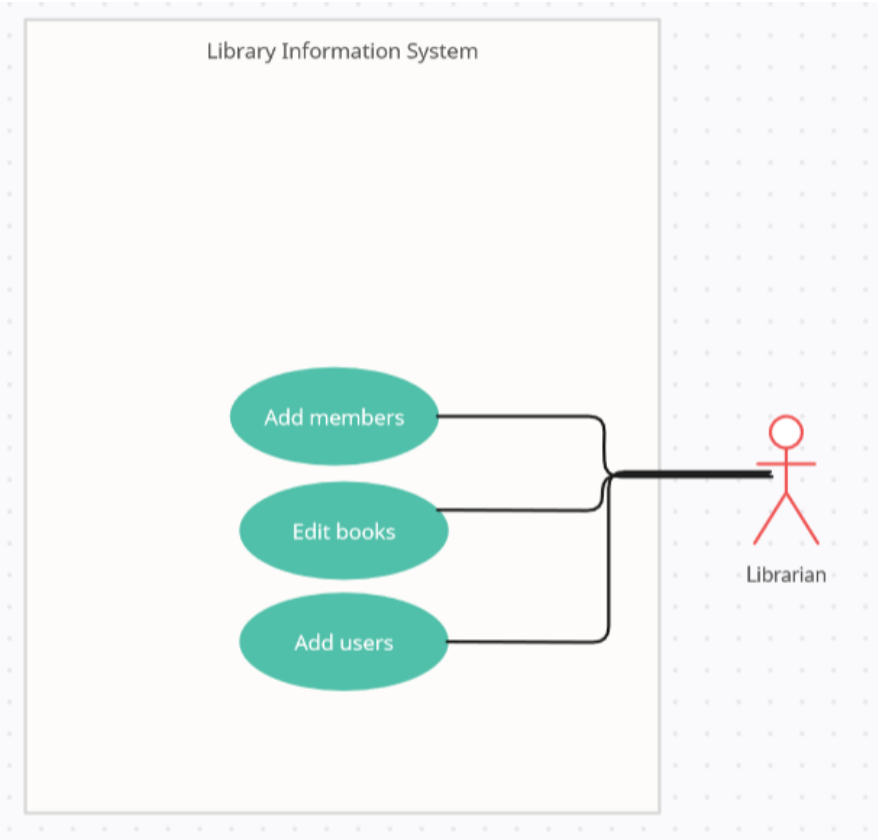
Use cases

- **Use Cases: A set of use cases that describe how the software will be used in different scenarios, along with any relevant system behaviors or interactions.**
- **Include the use case diagram**

I made two Use Diagrams. Not sure if they are right.



Describe the above UCD.



Example of a Use Case Diagram

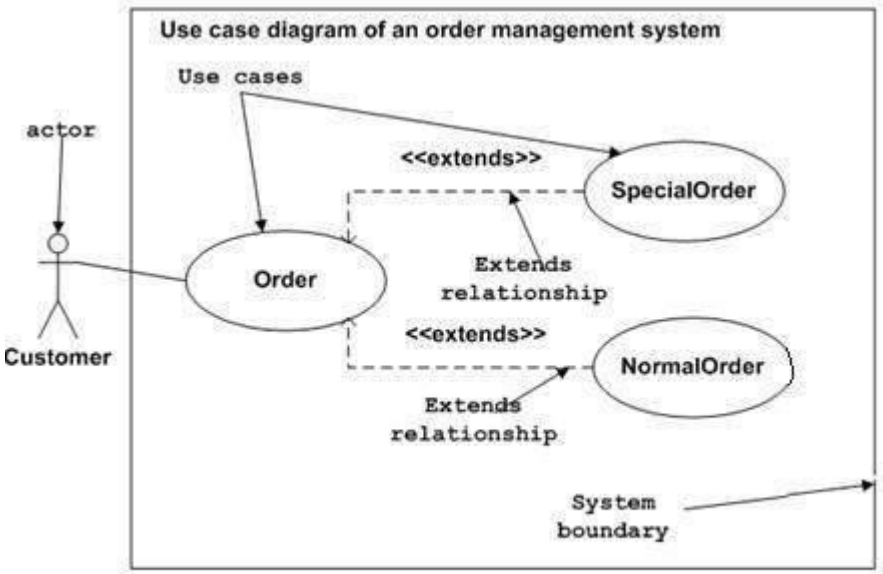


Figure: Sample Use Case diagram

Assumptions & Constraints

- **Assumptions and Constraints: Any assumptions or constraints that are relevant to the software's design or implementation.**

It is assumed that WPF (Windows Presentation Foundation) technology can meet the UI requirement and behaviour as per the Figma sketches. Where this is not the case, there will be a need to adapt to the technology's offering as becomes apparent from prototyping/experimentation.

It is assumed that there is a .NET connector for C# based database operations (CRUD operations: create, retrieve, update and delete). Also, the database system to be used is new to me and a minor learning curve has to be accommodated.

It is assumed that my existing I.T. skill sets will port adequately to the technical challenges and conditions that will evolve. An assumption is that the timetable in place for the development schedule will be adequately met and that existing I.T. skills will aid in meeting the deadlines apparent or pave the way for possible spare time.

Postgre database system note: It is envisaged that development shall be done with a cloud-based database system. This is due to working environment constraints, such as network security and restrictions on installing software on the class computers.

Acceptance Criteria

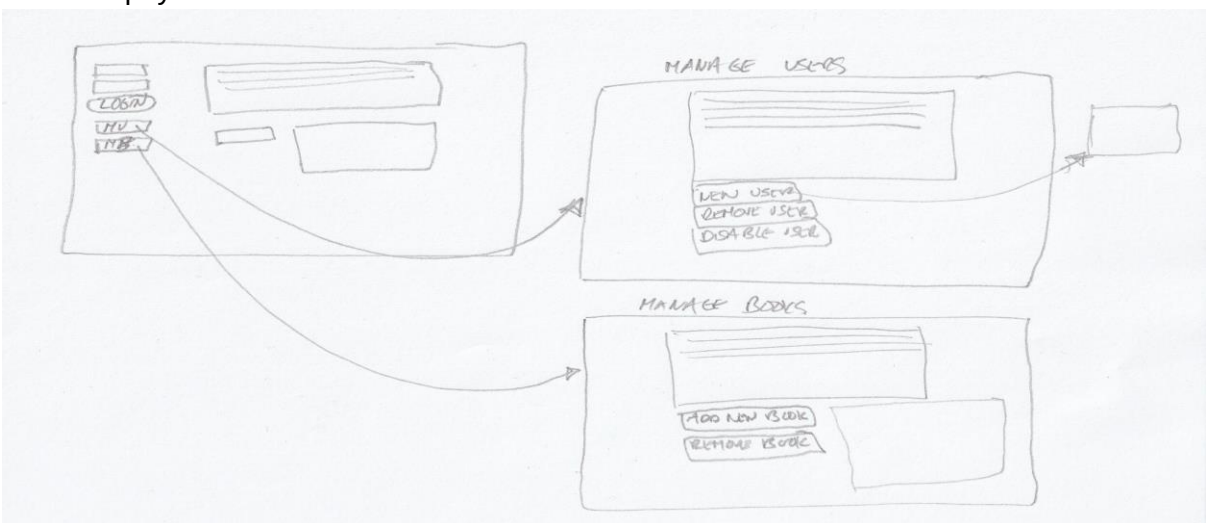
- Acceptance Criteria: Criteria that must be met for the software to be considered complete and acceptable to the stakeholders.(Formal approval by client/tutor)

The satisfactory implementation of all compulsory features and functionality. That the client is satisfied with ease of use features. For example, the size of the text on buttons and the general ease of use of the application.

Phase three - User interface design and prototyping

Sketches

Scans of physical sketches



The above is my hand drawn sketch of three windows. It is proposed. Top left is the Main screen. It features a login panel as well as a books browsing panel. Below the login panel are two buttons, only shown if the user is a librarian. One is for managing users, the other is for managing books.

The managing users button, when clicked, opens up a new window, featuring the user management user interface. Similarly, the managing books window, featuring the books management user interface.

Lo-Fi wireframes

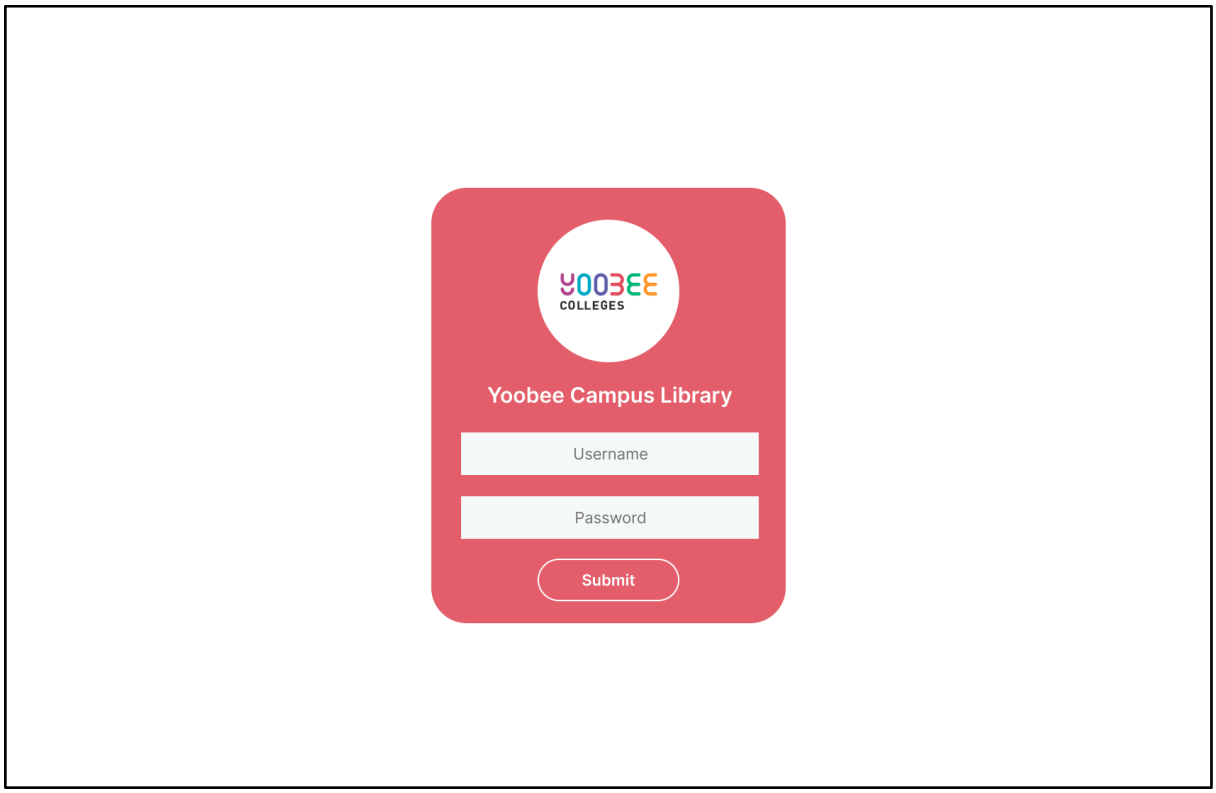
Screen Layout

Planning order and layout of navigation logic and flow

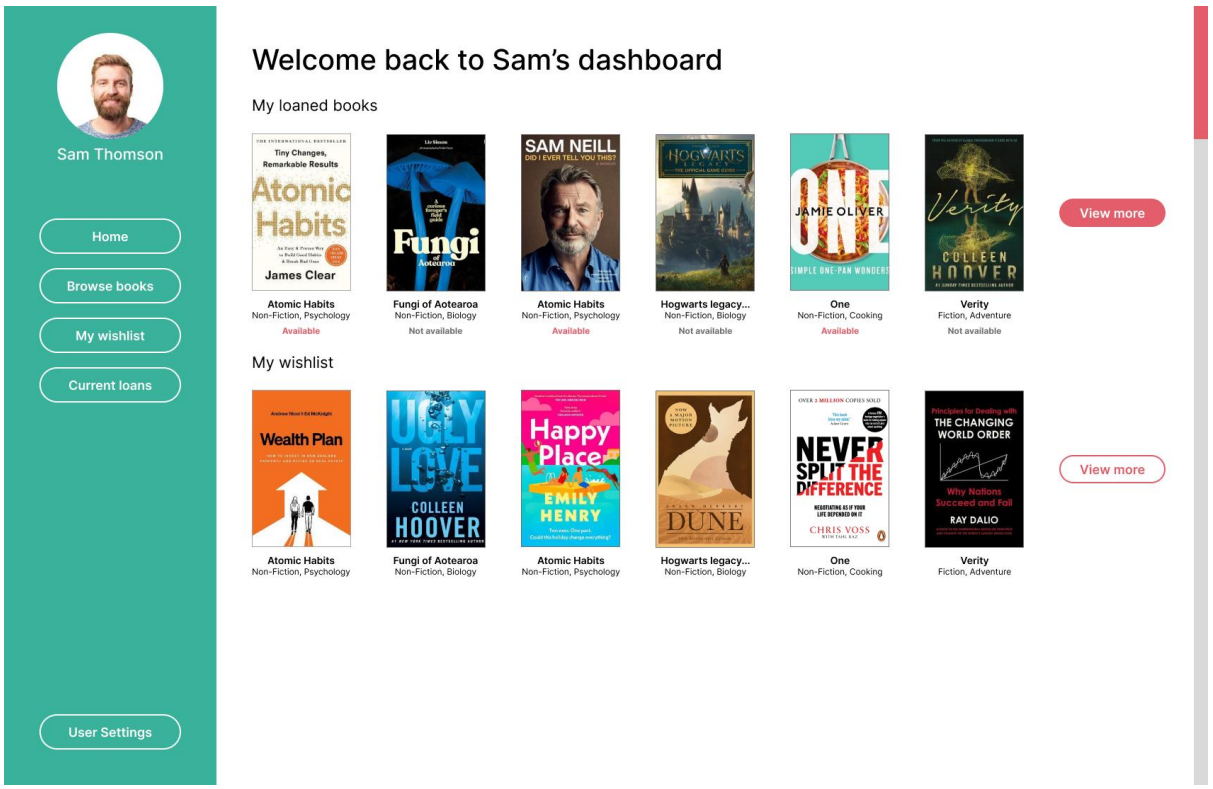
Hi-Fi prototype

Short description about our prototype

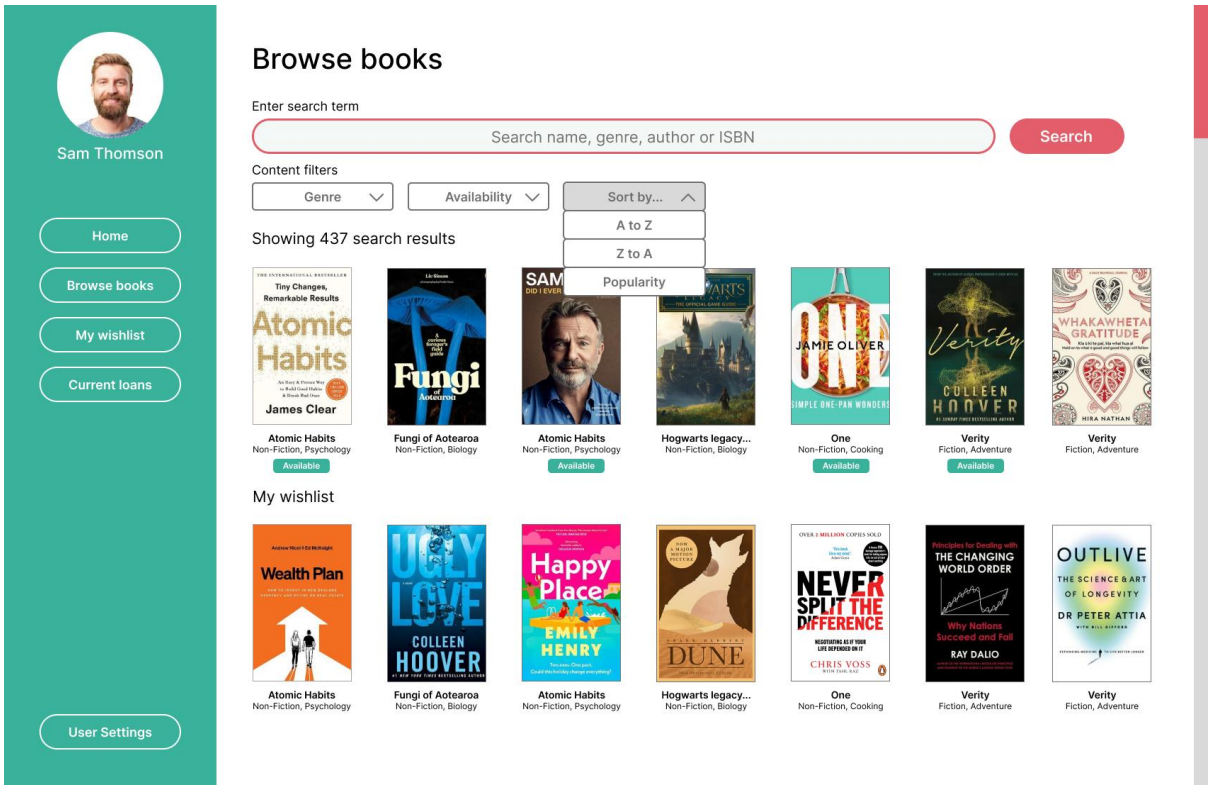
Log on page



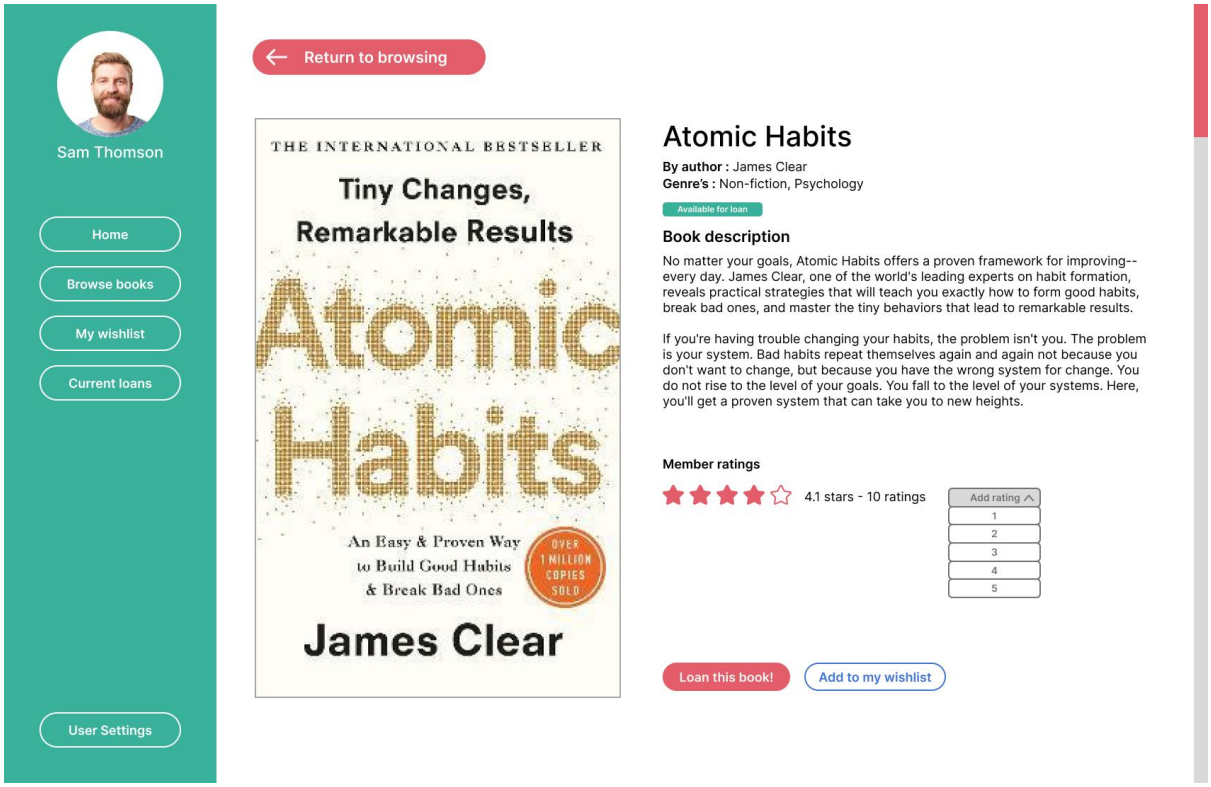
Member dashboard



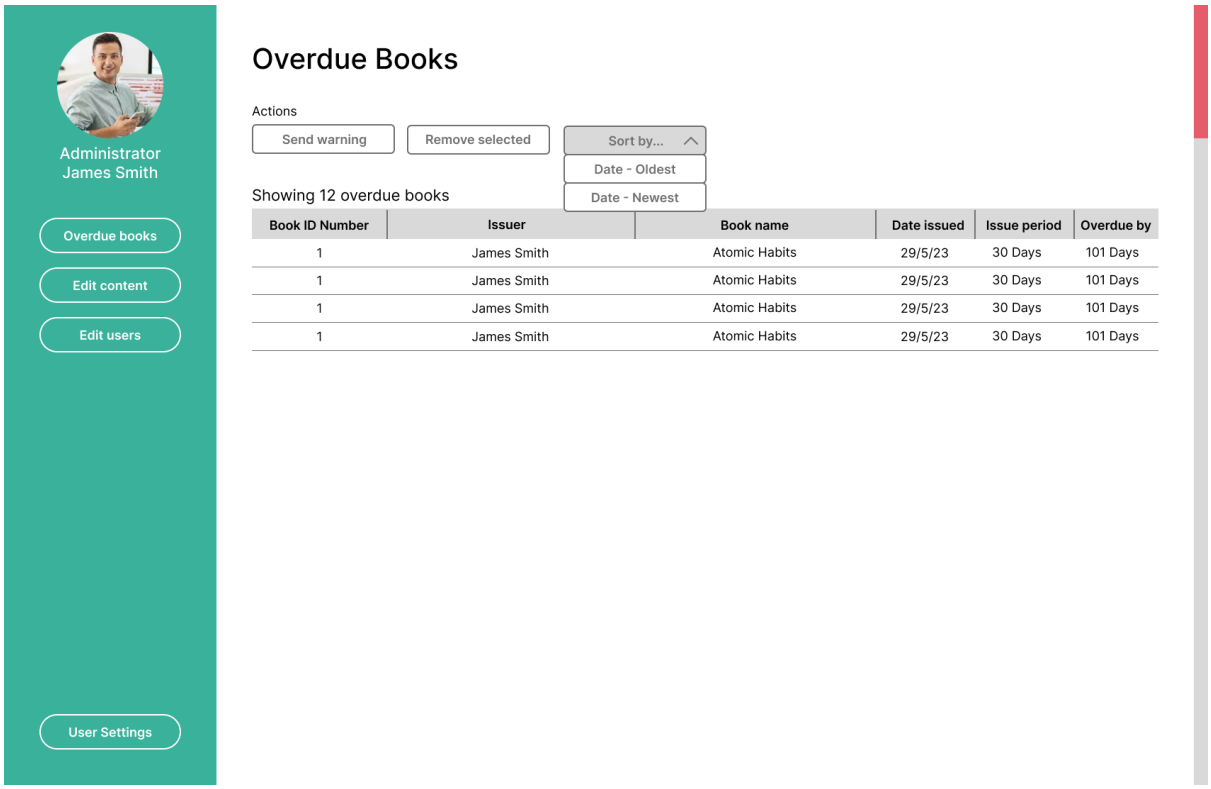
Member browse catalogue



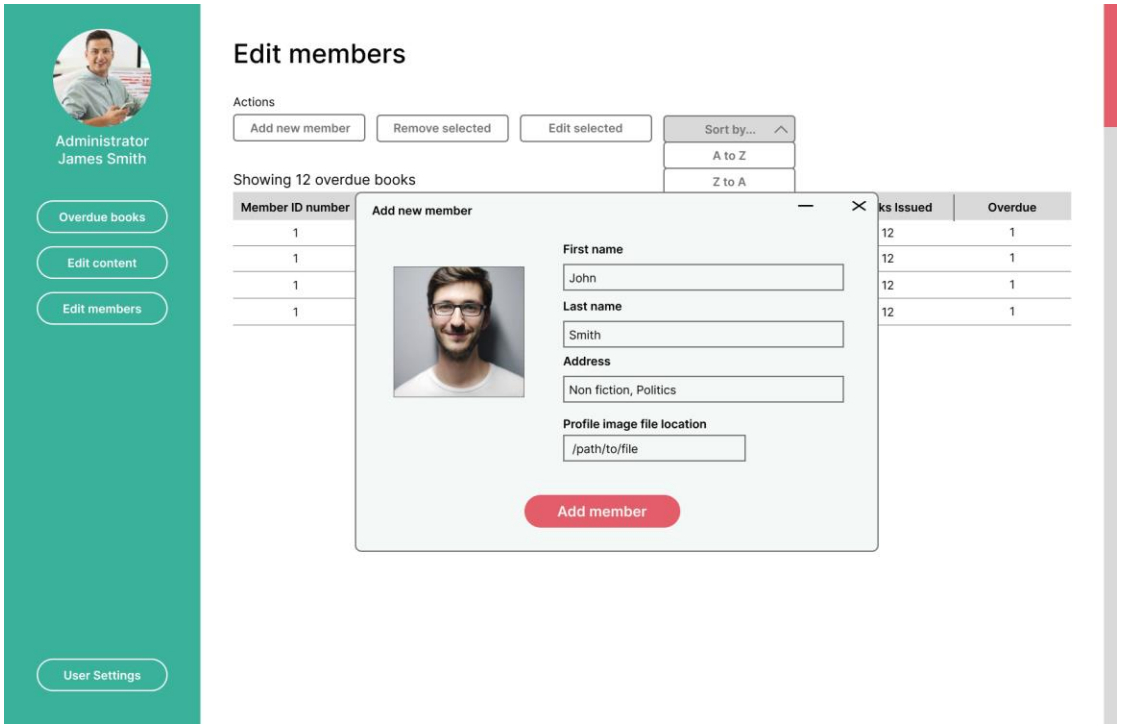
Examine, view selected book



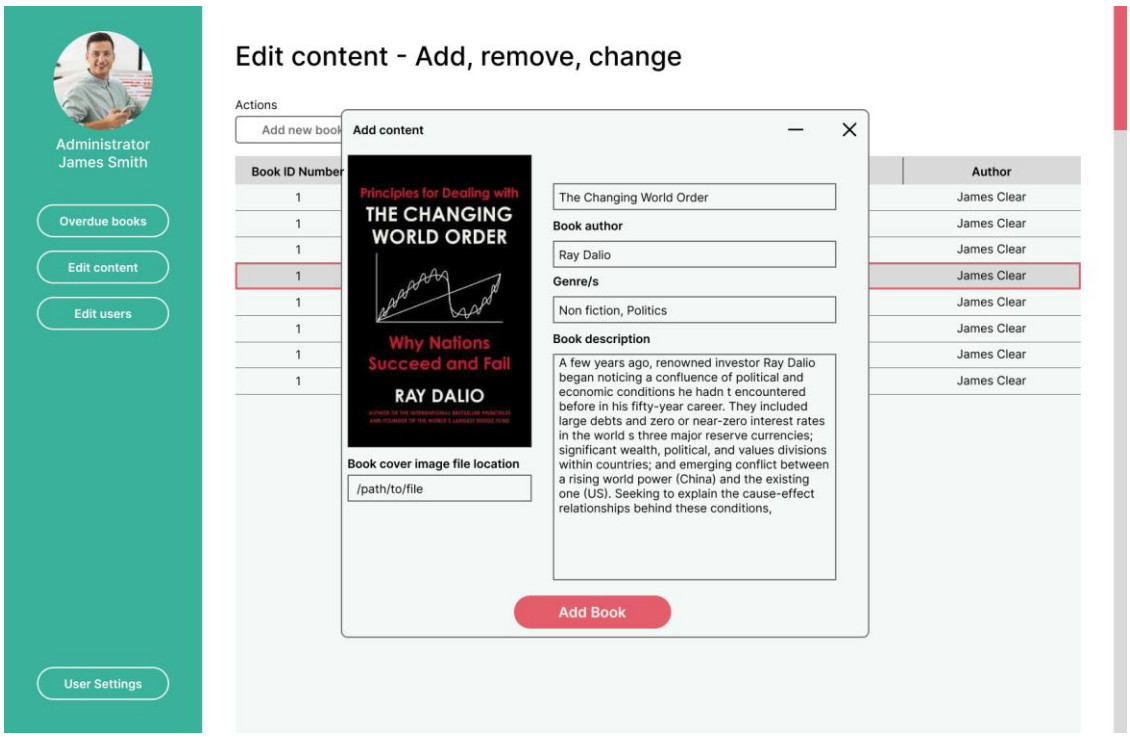
Administrator dashboard



Administrator - Edit members



Administrator - Edit book



User Testing to gain feedback for lo-fi wireframes and hi-fi prototypes